Subject: Re: Seemingly benign "Jekyll" app passes Apple review, then becomes "evil" | Ars Technica
From: "Phillip Shoemaker" [redacted]
Received(Date): Wed, 28 Aug 2013 16:15:10 +0000
To: "Craig Federighi" [redacted]
Cc: "Andreas Wendker" [redacted] ,"Phil Schiller" [redacted] , "Eddy Cue" [redacted] ,"Chris Lattner" [redacted]
Date: Wed, 28 Aug 2013 16:15:10 +0000

PLAINTIFF
U.S. District Court - NDCAL
4:20-cv-05640-YGR-TSH
Epic Games, Inc. v. Apple Inc.
Ex.No. PX-0465
Date Entered ____________
By ____________

Craig,

Thanks for the detailed response.

I agree that in the case of some of these apps, we will never be able to detect this behavior. However, if we truly believe that 15-20% of apps going into the store are guilty of using SPIs, shouldn't we fix this problem, Jekyll app or not?

The vast majority of these issues are simply because the developer knows one way around our tools: dlsym and dlopen. If we had a dynamic analyzer that could trace the true API being called through these methods, the majority of these violations will disappear. Most developers aren't clever enough to figure out the complex methods that Chris and others have come up with. In the various meetings we've had on this subject, the engineers have felt that they could implement something, providing that there was a tool that effectively exercised the app. We have such a tool with the Smoketest rack system that Eddy's team is working on.

Regarding the remote switch, we have only one idea, and one we haven't discussed with the tools team yet. Basically, capture usage logs when reviewing apps, and when we get logs from the wild, compare them on an app by app basis to see if they've fundamentally changed. If so, it triggers a re-review. Not sure how practical or useful this would be, but this is the only thing we've come up with.

The first step to solving this is to have a better static analyzer, or the beginnings of a dynamic analyzer that can more readily detect SPIs.

Thanks.

Phillip

On Aug 28, 2013, at 9:01 AM, Craig Federighi [redacted] wrote:

Phillip,

[cc'ing Phil, Eddy and Chris Lattner]

Exhibit
PX 465

I’ve followed up with Chris Lattner (head of our dev tools team) on this.

We recognize the value of solving this problem, and we’d love to have a solution.

The problem: this is a science project — we don’t know how to build something practical that won’t be rapidly worked around.

The issue is that even if we instrument sensitive APIs to see whether they’re getting called at runtime, we then also need to detect whether these calls are being made legitimately by our software, or whether the call is the work of the hacker. The hacker has many ways to synthesize call chains that would look to dynamic analysis like they came from our software, so we can’t determine with high confidence that the call is improper.

The problem is compound by the fact that any smart hacker will design their app to manifest this behavior only after the app has been approved and is running in the wild.

Fundamentally, though, the issue described in this article has nothing to do we detecting SPI use — it just has to do with the app changing behavior post review. Given that the review team can’t test how the app might react to triggers from data fetched from external servers, there’s no way for us to detect this issues a priori.

The only thing I can think that we could do would we to log back to Apple if we saw an unexpected pattern of calls to sensitive APIs. If we saw a statistically interesting boost in such activity, this would be a trigger for someone on the app review team to try to see whether this constituted a problem. Such data would be full of false positives, so screening them would likely be both highly technical and very resource intensive for the AppStore review team.

I wish that we had a better answer. Please let me know if you and your team have other ideas.

Thanks,

- craig

On Aug 27, 2013, at 3:13 PM, Phillip Shoemaker [redacted] wrote:

> Craig,
> Per the ERB email thread that went around earlier today, I'd like to ask you guys to consider implementing the dynamic analyzer that my team has requested these past few years. Only with a dynamic analyzer will we be able to find these so-called Jekyll apps, and the other 15-20% of apps that are using Private APIs but are currently flying under the radar.
>
> Begin forwarded message:
>
> ### Static Analyzer
>
> **Because Objective-C has calls like dlopen and dlsym, developers are able to dynamically open libraries on the fly, using APIs that are undetected using a**

static analyzer. This static analyzer looks at API names rather than true APIs being called, so there's often the issue of false positives. The Static Analyzer enables us to catch direct accessing of Private APIs, but it completely misses apps using indirect methods of accessing these Private APIs. This is what the authors used in their Jekyll apps.

A dynamic analyzer has been on the roadmap from DevTools for close to four years now, but doesn't appear to be getting any closer to being done. This is an important component in our ability to catch these types of apps, but more importantly, without this, I can guarantee that we will continue to allow apps accessing Private APIs to get onto the store, because we don't have the tools to catch these types of shenanigans.

A dynamic analyzer also needs the smoketest from the iTunes team, but we have no ETA on it's availability.

What is your thought about getting this approved with some bodies to focus on it? Phil and Eddy seem to be in support, but we really need engineering talent to help realize this dream.

Thanks in advance for your help.

Phillip Shoemaker
App Store Review

PX-0465.3